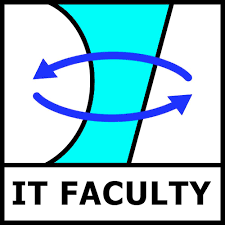
******ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----

****

**ĐỒ ÁN CƠ SỞ NGÀNH MẠNG**

**ĐỀ TÀI:**

**Phần Nguyên lí hệ điều hành*: Tìm hiểu* *cơ chế đa luồng và xây dựng trò chơi rắn săn mồi***

**Phần Lập trình mạng: *Xây dựng chương trình client tương tác với Web API***

**GVHD: PGS.TS Nguyễn Tấn Khôi**

**SVTH1: Nguyễn Phước Nhâm**

**MSSV: 102200222**

**Lớp: 20TCLC\_DT5**

**SVTH2: Nguyễn Trần Thảo Vy**

**MSSV: 102200121**

**Lớp: 20TCLC\_DT2**

***Đà Nẵng, 12/2022***

**NHẬN XÉT CỦA GIẢNG VIÊN**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

**…………………………………………………………………………………………..**

MỤC LỤC

[MỤC LỤC 3](#_Toc123142509)

[DANH MỤC HÌNH ẢNH 5](#_Toc123142510)

[PHẦN I: NGUYÊN LÝ HỆ ĐIỀU HÀNH 7](#_Toc123142511)

[CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI 7](#_Toc123142512)

[1. Nội dung đề tài 7](#_Toc123142513)

[2. Mục tiêu 7](#_Toc123142514)

[3. Môi trường sử dụng 7](#_Toc123142515)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 7](#_Toc123142516)

[1. Mở đầu 7](#_Toc123142517)

[2. Hệ điều hành windows 7](#_Toc123142518)

[1.1. 7](#_Toc123142520)

[2.1. Hệ điều hành windows 7](#_Toc123142521)

[2.2. Ưu điểm và nhược điểm của hệ điều hành windows 7](#_Toc123142522)

[3. Mạng máy tính 8](#_Toc123142523)

[3.1. Mạng máy tính là gì 8](#_Toc123142524)

[3.2. Giới thiệu chung về mạng máy tính 8](#_Toc123142525)

[3.3. Sự phát triển của mạng máy tính 9](#_Toc123142526)

[3.4. Hoạt động của mạng máy tính 9](#_Toc123142527)

[4. Mô hình Client – Server 10](#_Toc123142528)

[5. Mô hình truyền tin RMI (Remote Method Invocation) 12](#_Toc123142529)

[6. Cơ chế đa luồng trong java 15](#_Toc123142530)

[6.1. Tìm hiểu về cơ chế đa luồng trong java 15](#_Toc123142531)

[6.2. Tạo luồng bằng cách extend từ lớp Thread 17](#_Toc123142532)

[6.3. Tạo luồng bằng cách implement từ Interface Runnable 18](#_Toc123142533)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 19](#_Toc123142534)

[1. Giới thiệu về trò chơi rắn săn mồi 19](#_Toc123142535)

[2. Mô tả chương trình 19](#_Toc123142536)

[3. Thiết kế chương trình 20](#_Toc123142537)

[CHƯƠNG 4: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 21](#_Toc123142538)

[1. Giao diện 21](#_Toc123142539)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 29](#_Toc123142540)

[1. Kết luận 29](#_Toc123142541)

[1.1. Nhận xét 29](#_Toc123142542)

[2. Hướng phát triển 30](#_Toc123142543)

[PHẦN II: LẬP TRÌNH MẠNG 31](#_Toc123142544)

[CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI 31](#_Toc123142545)

[*1.* NỘI DUNG ĐỀ TÀI 31](#_Toc123142546)

[*2.* MỤC TIÊU 31](#_Toc123142547)

[3. MÔI TRƯỜNG SỬ DỤNG 31](#_Toc123142548)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 32](#_Toc123142549)

[1. REST API: 32](#_Toc123142550)

[1.1. Khái quát: 32](#_Toc123142551)

[1.2. Hai thành phần trong REST API 32](#_Toc123142552)

[1.3. Request và Response trong REST API 33](#_Toc123142553)

[1.4. Status Code 34](#_Toc123142554)

[2. HTTP/HTTPs 35](#_Toc123142555)

[2.1. Khái quát 35](#_Toc123142556)

[2.2. HTTP request và response 35](#_Toc123142557)

[3. Abp framework 36](#_Toc123142558)

[3.1. Khái quát 36](#_Toc123142559)

[3.2. Mô hình 36](#_Toc123142560)

[3.3. Application Modules 36](#_Toc123142561)

[4. Refit 37](#_Toc123142562)

[4.1. Khái quát 37](#_Toc123142563)

[4.2. Các dạng Request của Restful API 37](#_Toc123142564)

[4.3. Ưu điểm 37](#_Toc123142565)

[CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 38](#_Toc123142566)

[1. BÀI TOÁN VÀ YÊU CẦU 38](#_Toc123142567)

[1.1. Bài toán: 38](#_Toc123142568)

[*2.* THIẾT KẾ CHƯƠNG TRÌNH 38](#_Toc123142569)

[2.1. Server API 38](#_Toc123142570)

[2.2. Client- winform 47](#_Toc123142571)

[CHƯƠNG 4: TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ 50](#_Toc123142572)

[1. Giao diện 50](#_Toc123142573)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 58](#_Toc123142574)

[1. Ý nghĩa đề tài: 58](#_Toc123142575)

[1.1. Ưu điểm 58](#_Toc123142576)

[1.2. Nhược điểm 58](#_Toc123142577)

[2. Hướng phát triển: 58](#_Toc123142578)

[TÀI LIỆU THAM KHẢO 59](#_Toc123142579)

DANH MỤC HÌNH ẢNH

[Hình 1. Mô hình client - server 10](#_Toc123142313)

[Hình 2. Hoạt động mô hình client - server 11](#_Toc123142314)

[Hình 3. Mô hình RMI tổng quát 12](#_Toc123142315)

[Hình 4. Kiến trúc cơ bản của RMI 13](#_Toc123142316)

[Hình 5. Vòng đời các trạng thái của một thread trong java 19](#_Toc123142317)

[Hình 6. Giao diện server 21](#_Toc123142318)

[Hình 7. Khởi động server thành công 22](#_Toc123142319)

[Hình 8. Giao diện client 22](#_Toc123142320)

[Hình 9. Giao diện client khi chủ phòng tạo phòng thành công 23](#_Toc123142321)

[Hình 10. Giao diện client khi người chơi tham gia vào phòng của chủ phòng thành công 23](#_Toc123142322)

[Hình 11. Tạo phòng thành công những người chơi bắt đầu chơi 24](#_Toc123142323)

[Hình 12. Giao diện server khi người chơi đang chơi 24](#_Toc123142324)

[Hình 13. Giao diện bản đồ cổ điển 25](#_Toc123142325)

[Hình 14. Giao diện bản đồ BanDo01 25](#_Toc123142326)

[Hình 15. Giao diện bản đồ BanDo02 26](#_Toc123142327)

[Hình 16. Giao diện bản đồ BanDo03 26](#_Toc123142328)

[Hình 17. Giao diện bản đồ BanDo04 27](#_Toc123142329)

[Hình 18. Giao diện client của người chơi tkhi 4 người cùng tham gia vào một phòng 27](#_Toc123142330)

[Hình 19. Cảnh báo phòng đã đủ 4 người chơi 28](#_Toc123142331)

[Hình 20. Màn hình 4 người chơi cùng lúc trong phòng 28](#_Toc123142332)

[Hình 21. Khi người chơi tham gia vào phòng và nhập sai mã code 29](#_Toc123142333)

[Hình 1. Rest API – Giao tiếp client và server 32](#_Toc123142334)

[Hình 2. Quá trình chuyển email 33](#_Toc123142335)

[Hình 3. Ý nghĩa status code 34](#_Toc123142336)

[Hình 4. HTTP request và response 36](#_Toc123142337)

[Hình 5. Sơ đồ trình tự tổng quát 39](#_Toc123142338)

[Hình 6. Cấu trúc solution 39](#_Toc123142339)

[Hình 7. Class entity 40](#_Toc123142340)

[Hình 8. Class repository 40](#_Toc123142341)

[Hình 9. Class DTO 41](#_Toc123142342)

[Hình 10. ClassAppService 42](#_Toc123142343)

[Hình 11. File appsettings.json 43](#_Toc123142344)

[Hình 12. Xác định quyền truy cập 43](#_Toc123142345)

[Hình 13. Sử dụng các quyền truy cập ở appservice 44](#_Toc123142346)

[Hình 14. Tempalte class GetAPIByService để gọi API 47](#_Toc123142347)

[Hình 15. Get Token from server 47](#_Toc123142348)

[Hình 16. Interface trong SDK common 48](#_Toc123142349)

[Hình 17. Gọi API 48](#_Toc123142350)

[Hình 18. Xử lí ngoại lệ 49](#_Toc123142351)

[Hình 19. Appsetting server 50](#_Toc123142352)

[Hình 20. Appsetting local 50](#_Toc123142353)

[Hình 21. Giao diện login 51](#_Toc123142354)

[Hình 22. Giao diện chính 52](#_Toc123142355)

[Hình 23. Chi tiết, chỉnh sửa thông tin học sinh 53](#_Toc123142356)

[Hình 24. Chi tiết khoá học 53](#_Toc123142357)

[Hình 25. Chi tiết bài học 54](#_Toc123142358)

[Hình 26. Chi tiết lớp học 54](#_Toc123142359)

[Hình 27. Chi tiết buổi học 54](#_Toc123142360)

[Hình 28. Danh sách các API 55](#_Toc123142361)

[Hình 29. Thử gọi API trên swagger 55](#_Toc123142362)

[Hình 30. Kết quả 56](#_Toc123142363)

[Hình 31. Danh sách các vai trò 56](#_Toc123142364)

[Hình 32. Chỉnh sửa quyền cho vai trò 57](#_Toc123142365)

[Hình 33. Quản lí người dùng 57](#_Toc123142366)

1. NGUYÊN LÝ HỆ ĐIỀU HÀNH

Đề tài: Tìm hiểu cơ chế đa luồng và xây dựng trò chơi rắn săn mồi

# GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

## Nội dung đề tài

Tìm hiểu cơ chế đa luồng và xây dựng trò chơi rắn săn mồi cho nhiều người chơi trên nhiều client trên một máy tính

## Mục tiêu

Hiểu, nắm bắt được các khái niệm về luồng, đa luồng. Xây dựng được trò chơi theo cơ chế đa luồng với nhiều client

## Môi trường sử dụng

- Ngôn ngữ được sử dụng để viết chương trình là java

- Nền tảng lập trình là apache netbean, visual studio code

# CƠ SỞ LÝ THUYẾT

## Mở đầu

Ngày nay, với sự phát triển vượt bậc của công nghệ việc tìm cho mình một hình thức giải trí như game là hết sức dễ dàng và phổ biến, tựa game rắn săn mồi là tựa game cũ và không còn mới mẻ với mọi người. Nhưng trong đề tài lần này chúng em đã mang đến tựa game rắn săn mồi theo mô hình client-server trong mạn Lan cho phép nhiều người chơi sử dụng ngôn ngữ lập trình java trên nền tảng Apache NetBean để lập trình.

Đề tài lần này được chúng em sử dụng kỹ thuật RMI (Remote Method Invocation). Trong đề tài này chúng em chỉ chạy chương trình trình với nhiều người chơi trên một máy tính.

## Hệ điều hành windows



### 

### Hệ điều hành windows

Microsoft Windows (hoặc đơn giản là Windows) là tên của một họ hệ điều hành dựa trên giao diện người dùng đồ hoạ được phát triển và được phân phối bởi Microsoft. Nó bao gồm một vài các dòng hệ điều hành, mỗi trong số đó phục vụ một phần nhất định của ngành công nghiệp máy tính.

Giao diện của những hệ điều hành ban đầu cực kì đơn giản, tương tác với người dùng chủ yếu qua các dòng lệnh đơn thuần. Những phiên bản gần đây đã có sự cải thiện rõ nét: giao diện tương tác người dùng đồ họa đã bắt đầu phát triển.

### Ưu điểm và nhược điểm của hệ điều hành windows

* + Ưu điểm:
    - **Tính tương thích cao**

Vốn là một nền tảng chiếm thị phần sử dụng cao nhất hiện nay nên cũng không có gì khó hiểu khi hầu hết các nhà sản xuất đều đầu tư xây dựng phần mềm cũng như sản xuất phần cứng hỗ trợ cho hệ điều hành Windows.

* + - **Dễ sử dụng**

Các phiên bản hệ điều hành Windows tiếp theo luôn được kế thừa những tính năng cơ bản của phiên bản tiền nhiệm, giúp người dùng dễ dàng làm quen và sử dụng.

* + - **Bảo mật**

Dù không bảo mật toàn diện như: Linux, Mac OS ... nhưng Microsoft luôn cung cấp cho người dùng những gói nâng cấp và cập nhật miễn phí để tối ưu sự ổn định và khả năng bảo mật của thiết bị.

* + - **Phong phú ứng dụng**

Lượng lớn người dùng Windows luôn thu hút các nhà viết ứng dụng nên so với các hệ điều hành khác, lượng ứng dụng của Windows luôn ở mức phong phú cao hơn.

* + - **Hỗ trợ tối đa cho màn hình cảm ứng (Windows 8 trở lên)**

Từ phiên bản Windows 8 trở nên, nền tảng Windows đã được thiết kế để hỗ trợ tốt cho các thiết bị màn hình cảm ứng.

* + **Nhược điểm:**
    - Khi lượng người dùng đông đảo cũng là mục tiêu thu hút sự quan tâm của các tin tặc, hacker, ... Do đó, các virus, phần mềm gián điệp, mã độc... đều được viết để hoạt động trên nền tảng này.

## Mạng máy tính

### Mạng máy tính là gì

Mạng máy tính là cơ sở của giao tiếp trong CNTT. Chúng được sử dụng theo nhiều cách khác nhau và có thể bao gồm nhiều loại mạng khác nhau. Mạng máy tính là một tập hợp các máy tính được kết nối với nhau để chúng có thể chia sẻ thông tin. Mạng máy tính xuất hiện từ những năm 1960 và đã trải qua một chặng đường phát triển dài kể từ đó.

### Giới thiệu chung về mạng máy tính

Mạng máy tính là một số các máy tính được nối kết với nhau theo một cách nào đó nhằm mục đích để trao đổi chia sẻ thông tin cho nhau với những ưu điểm như sau:

• Nhiều người có thể dùng chung một một thiết bị ngoại vi (máy in, modem..),

một phần mềm.

• Dữ liệu được quản lý tập trung nên an toàn hơn, sự trao đổi thông tin dữ liệu

giữa những người dùng sẽ nhanh chóng hơn, thuận lợi hơn. Người dùng có thể

trao đổi thư tín với nhau một cách dễ dàng và nhanh chóng.

• Có thể cài đặt Internet trên một máy bất kỳ trong mạng, sau đó thiết lập, định

cấu hình cho các máy khác có thể thông qua máy đã được cài đặt chương trình

share Internet để cũng có thể kết nối ra Internet.

• Mang lại khả năng giao tiếp bằng email, video, nhắn tin nhanh và nhiều

phương pháp khác.

• Có thể chia sẻ file, phần mềm và chương trình điều hành trên các hệ thống từ

xa.

### Sự phát triển của mạng máy tính

Mạng ngày nay cung cấp nhiều thứ hơn là kết nối. Các tổ chức đang bắt tay vào chuyển đổi kỹ thuật số. Mạng đóng vai trò rất quan trọng đối với sự chuyển đổi này và thành công của các tổ chức. Các loại kiến trúc mạng đang phát triển để đáp ứng những nhu cầu này bao gồm:

Do phần mềm xác định (Software-defined Network - SDN): Để đáp ứng các yêu cầu mới trong thời đại "kỹ thuật số", kiến trúc mạng đang phát triển theo hướng có thể lập trình, tự động và mở hơn. Trong các mạng do phần mềm xác định, việc định tuyến lưu lượng được điều khiển tập trung thông qua các cơ chế dựa trên phần mềm. Điều này giúp mạng phản ứng nhanh chóng với các điều kiện thay đổi.

Dựa trên mục đích (Intent-based Network - IBN): Xây dựng dựa trên các nguyên tắc SDN, mạng dựa trên mục đích không chỉ mang lại sự nhanh chóng, mà còn thiết lập một mạng để đạt được những mục tiêu mong muốn, bằng cách tự động hóa các hoạt động một cách rộng rãi, phân tích hiệu suất của nó, xác định các khu vực có vấn đề, cung cấp bảo mật toàn diện và tích hợp với nhiều quy trình kinh doanh.

Ảo hóa: Cơ sở hạ tầng mạng vật lý bên dưới có thể được phân vùng một cách hợp lý, để tạo ra nhiều mạng "lớp phủ". Mỗi mạng logic này có thể được điều chỉnh để đáp ứng các yêu cầu cụ thể về bảo mật, chuẩn QoS và nhiều yêu cầu khác.

Dựa trên bộ điều khiển: Network controller (bộ điều khiển mạng) rất quan trọng đối với việc mở rộng và bảo mật mạng. Bộ điều khiển tự động hóa các chức năng mạng bằng cách chuyển mục đích kinh doanh sang cấu hình thiết bị và chúng giám sát thiết bị liên tục để giúp đảm bảo hiệu suất và bảo mật. Bộ điều khiển đơn giản hóa hoạt động và giúp tổ chức đáp ứng các yêu cầu kinh doanh thay đổi.

Tích hợp đa miền: Các doanh nghiệp lớn hơn có thể xây dựng các mạng riêng biệt, còn được gọi là networking domain, cho văn phòng, mạng WAN và trung tâm dữ liệu của mình. Các mạng này giao tiếp với nhau thông qua bộ điều khiển của chúng. Những tích hợp liên mạng hoặc đa miền như vậy thường liên quan đến việc trao đổi các thông số hoạt động có liên quan để giúp đảm bảo đạt được những kết quả kinh doanh mong muốn trên các domain mạng.

### Hoạt động của mạng máy tính

Các thiết bị chuyên dụng như thiết bị switch, router và access point (điểm truy cập) tạo thành nền tảng của mạng máy tính.

Switch kết nối và giúp bảo mật nội bộ máy tính, máy in, máy chủ và các thiết bị khác với mạng trong gia đình hoặc tổ chức. Điểm truy cập là switch kết nối thiết với mạng mà không cần sử dụng dây cáp.

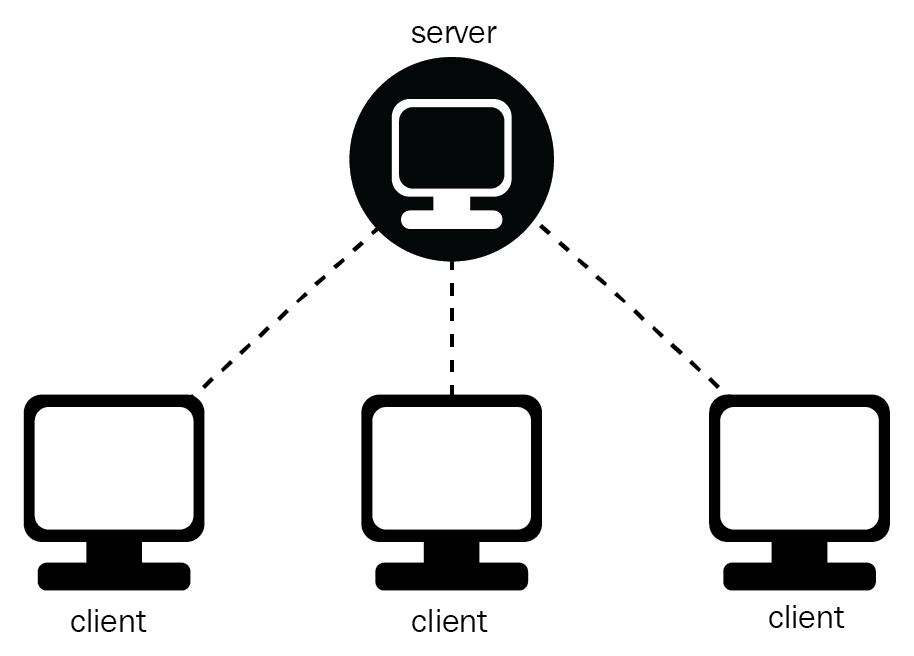
Router (bộ định tuyến) kết nối mạng này với các mạng khác và hoạt động với vai trò điều phối. Các thiết bị này phân tích dữ liệu được gửi qua một mạng, chọn các tuyến đường tốt nhất cho nó và gửi nó đến vị trí đích. Router kết nối ngôi nhà và doanh nghiệp của bạn với thế giới, cũng như giúp bảo vệ thông tin khỏi các mối đe dọa bảo mật bên ngoài.

Mặc dù switch và router khác nhau ở một số khía cạnh, nhưng một điểm khác biệt chính là cách chúng xác định thiết bị cuối. Switch Layer 2 xác định duy nhất một thiết bị bằng địa chỉ MAC "đã ghi sẵn" của nó. Router Layer 3 xác định duy nhất kết nối mạng của thiết bị bằng địa chỉ IP được chỉ định bởi mạng. Ngày nay, hầu hết các thiết bị switch đều bao gồm một số cấp độ chức năng định tuyến. Địa chỉ MAC và IP tương ứng xác định duy nhất các thiết bị và kết nối mạng trong một mạng. Địa chỉ MAC là một số được nhà sản xuất thiết bị gán cho interface card mạng (Network Interface Card- NIC). Địa chỉ IP là một số được gán cho kết nối mạng.

Mạng máy tính ngày nay cung cấp nhiều thứ khác ngoài khả năng kết nối. Các tổ chức đang bắt tay vào chuyển đổi kỹ thuật số, trong đó mạng máy tính đóng vai trò rất quan trọng đối với sự chuyển đổi này và thành công của doanh nghiệp.

## Mô hình Client – Server

Mô hình Client Server là mô hình mạng máy tính trong đó các máy tính con được đóng vai trò như một máy khách, chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ. Để máy chủ xử lý yêu cầu và trả kết quả về cho máy khách đó.



1. Mô hình client - server

+ Mô hình client/server cung cấp một cách tiếp cận tổng quát để chia sẻ tài nguyên trong các hệ thống phân tán.

+ Cả tiến trình client và tiến trình server đều có thể chạy trên cùng một máy tính.

+ Một tiến trình server có thể sử dụng dịch vụ của một server khác.

+ Mô hình truyền tin client/server hướng tới việc cung cấp dịch vụ.

Quá trình trao đổi dữ liệu bao gồm:

- Bước 1: Truyền một yêu cầu từ tiến trình client tới tiến trình server

- Bước 2: Yêu cầu được server xử lý

- Bước 3: Truyền đáp ứng cho client

Mô hình client/server cung cấp một cách tiếp cận tổng quát để chia sẻ tài nguyên trong các hệ thống phân tán. Mô hình này có thể được cài đặt bằng rất nhiều môi trường phần cứng và phần mềm khác nhau. Các máy tính được sử dụng để chạy các tiến trình client/server có nhiều kiểu khác nhau và không cần thiết phải phân biệt giữa chúng; cả tiến trình client và tiến trình server đều có thể chạy trên cùng một máy tính. Một tiến trình server có thể sử dụng dịch vụ của một server khác.

Mô hình truyền tin này liên quan đến việc truyền hai thông điệp và một dạng đồng bộ hóa cụ thể giữa client và server. Tiến trình server phải nhận thức được thông điệp được yêu cầu ở bước một ngay khi nó đến và hành động phát ra yêu cầu trong client phải được tạm dừng (bị phong tỏa) và buộc tiến trình client ở trạng thái chờ cho tớ khi nó nhận được đáp ứng do server gửi về ở bước ba.

Mô hình client/server thường được cài đặt dựa trên các thao tác cơ bản là gửi (send) và nhận (receive)

Timeline

Description automatically generated

1. Hoạt động mô hình client - server

Quá trình giao tiếp client và server có thể diễn ra theo một trong hai chế độ: bị phong tỏa (blocked) và không bị phong tỏa (non-blocked):

• Chế độ bị phong tỏa (blocked):

• Trong chế độ bị phong tỏa, khi tiến trình client/server phát ra lệnh gửi dữ liệu (send), việc thực thi của tiến trình sẽ bị tạm ngừng cho tới khi tiến trình nhận phát ra lệnh nhận dữ liệu (receive).

• Tương tự đối với tiến trình nhận dữ liệu, nếu tiến trình nào đó (client \*\*\* server) phát ra lệnh nhận dữ liệu, mà tại thời điểm đó chưa có dữ liệu gửi tới thì việc thực thi của tiến trình cũng sẽ bị tạm ngừng cho tới khi có dữ liệu gửi tới.

• Chế độ không bị phong tỏa (non-blocked) :

• Trong chế độ này, khi tiến trình client hay server phát ra lệnh gửi dữ liệu thực sự, việc thực thi của tiến trình vẫn được tiến hành mà không quan tâm đến việc có tiến trình nào phát ra lệnh nhận dữ liệu đó hay không.

• Tương tự cho trường hợp nhận dữ liệu, khi tiến trình phát ra lệnh nhận dữ liệu, nó sẽ nhận dữ liệu hiện có, việc thực thi của tiến trình vẫn được tiến hành mà không quan tâm đến việc có tiến trình nào phát ra lệnh gửi dữ liệu tiếp theo hay không.

## Mô hình truyền tin RMI (Remote Method Invocation)

RMI - Remote Method Invocation là một kĩ thuật cài đặt các đối tượng phân tán trong Java. RMI là một phần của bộ J2SDK và là hàm thư viện hỗ trợ các lời gọi phương thức từ xa và trả về giá trị cho các ứng dụng tính toán phân tán. Chúng ta giả sử rằng ngôn ngữ Java được sử dụng ở cả hai phía gọi và phía bên phương thức được gọi.

Kỹ thuật RMI - mang ý nghĩa là triệu gọi phương thức từ xa là cách thức giao tiếp giữa các đối tượng trong Java có mã lệnh cài đặt nằm ở trên các máy khác nhau có thể triệu gọi lẫn nhau. Để giải quyết một số vấn đề trong việc truyền thông giữa client - server. RMI không gọi trực tiếp mà thông qua lớp trung gian.

Diagram

Description automatically generated

1. Mô hình RMI tổng quát

Thông thường các chương trình của chúng ta được viết dưới dạng thủ tục hàm và việc các hàm gọi lẫn nhau và truyền tham số chỉ xảy ra ở máy cục bộ. Kỹ thuật RMI - mang ý nghĩa là triệu gọi phương thức từ xa là cách thức giao tiếp giữa các đối tượng trong Java có mã lệnh cài đặt nằm ở trên các máy khác nhau có thể triệu gọi lẫn nhau.

Để giải quyết một số vấn đề trong việc truyền thông giữa Client - Server. RMI không gọi trực tiếp mà thông qua lớp trung gian. Lớp này tồn tại ở cả 2 phía client và server.

- Lớp Client gọi là Stub

- Lớp ở máy Server gọi là Skel(Skeleton)

+ Các đặc tính của RMI

RMI là mô hình đối tượng phân tán của Java, nó giúp cho việc truyền thông giữa các đối tượng phân tán được dễ dàng hơn

RMI được xây dựng trên các socket, nếu không có socket RMI sẽ không thể hoạt động được

RMI là API bậc cao được xây dựng dựa trên việc lập trình socket

RMI registry: Bộ đăng kí này sẽ đăng kí một Remote object với Naming Registry. Giúp các Remote object được chấp nhận khi gọi các method từ xa

RMI không những cho phép chúng ta truyền dữ liệu giữa các đối tượng trên các hệ thống máy tính khác nhau và còn gọi được các phương thức trong các đối tượng ở xa.

Việc truyền dữ liệu giữa các máy khác nhau được sử lý một cách trong suốt bởi máy ảo Java(Java virtual machine)

RMI cung cấp cơ chế callback, nó cho phép Server triệu gọi các phương thức ở Client

Diagram

Description automatically generated

1. Kiến trúc cơ bản của RMI

+ Truyền tin trong RMI:

- RMI sử dụng lớp trung gian để truyền tin Skeleton và Stub

- Lớp Stub dùng ở client

- Lớp Skeleton dùng ở Server

- Java sử dụng rmic.exe để tạo ra các lớp trung gian

- TCP Socket

+ Cách thức hoạt động của RMI:

- Server RMI phải đăng ký với 1 dịch vụ tra tìm và đăng ký tên miền.

- Sau khi server được đăng ký, nó sẽ chờ các yêu cầu của RMI client.

- Các client RMI sẽ gửi thông điệp RMI để gọi một phương thức trên một đối tượng từ xa.

- Ứng dụng client yêu cần một tên dịch vụ cụ thể và nhận một URL trỏ đến tài nguyên từ xa.

+ Hướng dẫn cài đặt một chương trình có sử dụng RMI

- Tạo một server của chương trình

- Tạo một client của chương trình

- Ta sẽ định nghĩa một Interface dùng chung Interface này như là một thông điệp cho cả Server và Client, cả client và server đều phải tuân thủ nó, chỉ được triển khai và gọi các phương thức cho phép. Các phương thức đều kèm theo throws RemoteException.

**Cách chạy một ứng dụng java RMI(Remode method Invocation):**

Stub: stub là một đối tượng, hoạt động như một cổng ở phía máy khách. Tất cả các yêu cầu gửi đi đều được chuyển qua nó. Nó cư trú tại phía Client và đại diện cho đối tượng remote. Khi người gọi gọi phương thức trên đối tượng stub, nó sẽ thực hiện các công việc sau:

1. Khởi tạo kết nối với máy ảo từ xa (JVM)

2. Ghi và truyền các tham số tới máy ảo từ xa (JVM)

3. Đợi kết quả trả về

4. Đọc giá trị trả về hoặc các ngoại lệ (exception)

5. Cuối cùng, trả về giá trị cho người gọi.

Skeleton: là một đối tượng, hoạt động như một cổng cho đối tượng phía máy chủ. Tất cả các yêu cầu đến được chuyển qua nó. Khi skeleton nhận được yêu cầu tới, nó sẽ thực hiện các công việc sau:

1. Đọc các tham số cho phương thức remote

2. Goi phương thức trên đối tượng remote thực tế

3. Viết và truyền kết quả tới người gọi.

**Để viết một ứng dụng với Java RMI, bạn cần phải theo các bước sau:**

* + - Định nghĩa một lớp giao diện (Remote Interface)
    - Triển khai lớp thực thi của lớp giao diện (remote object)
    - Viết code cho chương trình chạy ở Server
    - Viết code cho chương trình chạy ở Client
    - Biên dịch ứng dụng
    - Chạy ứng dụng

## Cơ chế đa luồng trong java

### Tìm hiểu về cơ chế đa luồng trong java

Thread (luồng) về cơ bản là một tiến trình con (sub-process). Một đơn vị xử lý nhỏ nhất của máy tính có thể thực hiện một công việc riêng biệt. Trong Java, các luồng được quản lý bởi máy ảo Java (JVM).

Process (tiến trình) là một chương trình chạy trên hệ điều hành và được quản lý thông qua các thẻ. Tiến trình là một đơn vị xử lý cơ bản của hệ thống, một tiến trình sở hữu nhiều tiểu trình.

Một tiến trình có thể bao gồm nhiều luồng. Các luồng của một tiến trình có thể chia sẻ với nhau về không gian địa chỉ chương trình, các đoạn dữ liệu và môi trường xử lý, đồng thời cũng có vùng dữ liệu riêng để thao tác.

Đa nhiệm: ở cùng một thời điểm có nhiều hơn một tiến trình thực hiện đồng thời trên cùng một máy tính. Có 2 kỹ thuật đa nhiệm:

- Đa nhiệm dựa trên các tiến trình

- Đa nhiệm dựa trên các luồng

Kỹ thuật đa nhiệm cho phép tận dụng được những thời gian rỗi của CPU để thực hiện những tác vụ khác.

Đa nhiệm được thực hiện theo 2 cách:

- Phụ thuộc và hệ điều hành, nó có thể cho tạm ngừng chương trình mà không cần tham khảo các chương trình đó.

- Các chương trình chỉ bị dừng lại khi chúng tự nguyện nhường điều khiển cho chương trình khác.

Nhiều hệ điều hành hiện nay hổ trợ đa luồng, java hổ trợ đa nhiệm dựa trên các luồng và cung cấp đặc tính ở mức cao cho lập trình đa luồng.

**Tạo và quản lý luồng:**

- Khi chương trình java thực thi thì hàm main() tức là luồng main được thực thi.

- Tuyến này được tạo ra một cách tự động, tại đây:

- Các luồng con sẽ được tạo ra từ đó

- Nó là luồng cuối cùng kết thúc thực thi. Ngay khi luồng main() ngừng thực thi, chương trình bị chấm dứt.

**Phân chia thời gian giữa các luồng:**

CPU thực thi chỉ một luồng tạ một thời điểm nhất định.

Cấc luồng có độ ưu tiên bằng nhau thì được phân chia thời gian sử dụng bộ vi xử lý.

**Lập trình đa luồng:**

Với Java có thể xây dựng các chương trình đa luồngl

Một ứng dụng có thể bao gồm nhiều luông, mỗi luồng được gắn công việc cụ thể và được thực thi đồng thời với các luông khác.

**Java cung cấp hai giải pháp tạo lập luồng**:

- Thiết lập lớp con thread

- Cài đặt xử lý luồng từ giao diện Runnable

**Các trạng thái và phương thức của lớp Thread**

 Trạng thái:

* + - born
    - ready to run
    - running
    - sleeping
    - waiting
    - ready
    - blocked
    - dead

 Phương thức:

* + - start( )
    - sleep( )
    - wait( )
    - notify( )
    - run( )
    - stop( )

**Các trạng thái của Thread**

+ New: Khi một luồng mới được tạo ra với toán tử new() và sẵn sàng hoạt động

+ Runnable: Trạng thái mà luồng đang chiếm CPU để thực hiện, khi bắt đầu thì nó gọi hàm start(). Bộ lập lịch phân luồng của hệ điều hành sẽ quyết định luồng nào sẽ được chuyển về trạng thái Runnable và hoạt động. Cũng cần lưu ý rằng ở một thời điểm, một luồng ở trạng thái Runnable có thể hoặc không có thể thực hiện.

+ Non runnable (blocked): Từ trạng thái runnable chuyển sang trạng thái ngừng thực hiện (“bị chặn”) khi gọi một trong các hàm: sleep(), suspend(), wait(), hay bị chặn lại ở input/output.

- Một luồng có thể ở một trong các trạng thái sau:

+ Waitting: Khi ở trạng thái Runnable, một luồng thực hiện hàm wait() thì nó chuyển sang trạng thái đợi chờ (waitting).

+ Sleeping(): Khi ở trạng thái Runnable, một luồng thực hiện hàm sleep() thì nó sẽ chuyển sang trạng thái ngủ (sleeping).

+ Blocked(): Khi ở trạng thái Runnable, một luồng bị chặn lại bởi những yêu cầu về tài nguyên, như yêu cầu vào/ra (I/O), thì nó sẽ chuyển sang trạng thái bị chặn (blocked).

- Mỗi luồng phải thoát ra khỏi trạng thái Blocked để quay về trạng thái Runnable, khi:

+ Nếu một luồng đã được cho đi “ngủ” (sleep) sau khoảng thời gian bằng số micro giây n đã được truyền vào tham số của hàm sleep(n).

+ Nếu một luồng bị chặn lại vì vào/ra và quá trình này đã kết thúc.

+ Nếu luồng bị chặn lại khi gọi hàm wait(), sau đó được thông báo tiếp tục bằng cách gọi hàm notify() hoặc notifyAll().

+ Nếu một luồng bị chặn lại để chờ monitor của đối tượng đang bị chiếm giữ bởi luồng khác, khi monitor đó được giải phóng thì luồng bị chặn này có thể tiếp tục thực hiện (khái niệm monitor được đề cập ở phần sau).

+ Nếu một luồng bị chặn lại bởi lời gọi hàm suspend(), muốn thực hiện thì trước đó phải gọi hàm resume().

+ Hàm suspend() có tác dụng tạm ngừng tuyến, ít được dùng do không nhả tài nguyên của hệ thống, dễ dẫn đến deaklock. Nếu ta gọi các hàm không phù hợp đối với các luồng thì JVM sẽ phát sinh ra ngoại lệ IllegalThreadStateException.

+ Dead: Luồng chuyển sang trạng thái “chết” khi nó kết thúc hoạt động bình thường, hoặc gặp phải ngoại lệ không thực hiện tiếp được.

+ Trong trường hợp đặc biệt, bạn có thể gọi hàm stop() để kết thúc (“giết chết”) một luồng

### Tạo luồng bằng cách extend từ lớp Thread

- Để tạo luồng bằng cách tạo lớp kế thừa từ lớp Thread, ta phải làm các công việc sau :

1.Khai báo 1 lớp mới kế thừa từ lớp Thread

2.Override lại phương thức run ở lớp này, những gì trong phương thức run sẽ được thực thi khi luồng bắt đầu chạy. Sau khi luồng chạy xong tất cả các câu lệnh trong phương thức run thì luồng cũng tự hủy.

3.Tạo 1 thể hiện (hay 1 đối tượng) của lớp ta vừa khai báo.

4.Sau đó gọi phương thức start() của đối tượng này để bắt đầu thực thi luồng.

- Ví dụ: Tạo một lớp kế thừa từ lớp Thread và ghi đè phương thức của lớp Thread như sau:

Class MyClass extends Thread

{

//một số thuộc tính

Public void run()

{

// các lệnh cần thực hiện theo luồng

}

// một số hàm khác được viết đè hay được bổ sung

}

=> Khi chương trình chạy nó sẽ gọi hàm đặc biệt đã được khai báo trong Thread đó là start().

Để bắt đầu một luồng đã được tạo ra.

### Tạo luồng bằng cách implement từ Interface Runnable

- Để tạo luồng bằng cách hiện thực từ Interface Runnable, ta phải làm các công việc sau:

1.Khai báo 1 lớp mới implements từ Interface Runnable

2.Hiện thực phương thức run() ở lớp này, những gì trong phương thức run() sẽ được thực thi khi luồng bắt đầu chạy. Sau khi luồng chạy xong tất cả các câu lệnh trong phương thức run thì luồng cũng tự hủy.

3.Tạo 1 thể hiện (hay 1 đối tượng) của lớp ta vừa khai báo. (VD : Tên đối tượng là r1)

4.Tạo 1 thể hiện của lớp Thread bằng phương thức khởi tạo : Thread(Runnable target)

- Runnable target: Là 1 đối tượng thuốc lớp được implements từ giao diện Runnable.

- Ví dụ: Thread t1 = new Thread(r1);

5.Gọi phương thức start() của đối tượng t1.

- java giải quyết hạn chế trên bằng cách xây dựng lớp để tạo ra các luồng thực hiện trên cơ sở

Cài đặt giao diện hổ trợ luồng

- Tạo ra một lớp triển khai từ giao diện Runnable, cài đặt phương thức run

- ví dụ:

Class MyClass implements Runnable {

//các thuộc tính

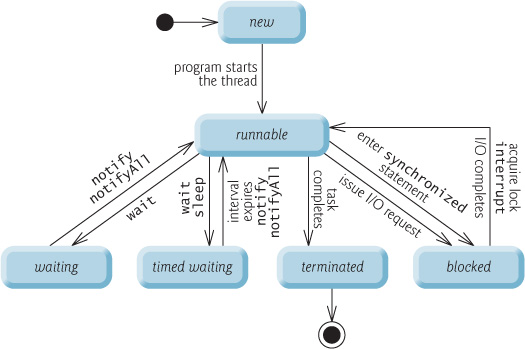
//nạp chồng hay viết đè một số hàm

Public void run{

....

}

}



1. Vòng đời các trạng thái của một thread trong java

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Giới thiệu về trò chơi rắn săn mồi

Rắn săn mồi là một trò chơi không còn mới, rất phổ biến và đã xuất hiện từ lâu, được xây dựng, cải tiến và làm mới không ngừng. Cho đến ngày nay vẫn được rất nhiều người chơi yêu thích.

Trong đề tài lần này chúng em xây dựng game rắn săn mồi có thể chơi một người và cũng có thể chơi nhiều người trên cùng một máy tính.

## Mô tả chương trình

- Viết chương trinh cho Server luôn ở trạng thái running đợi Client (người chơi) kết nối vào.

- Viết chương trình cho Client, cho phép kết nối đến Server thông qua IP

- Người chơi (Client) kết nối vào game.

- Người chơi tạo ra phòng chơi là chủ phòng

- Người chơi vào phòng chơi đã tạo trước thông qua mã code 4 chữ số, (mã này chủ phòng tự thông báo cho những người khác, hệ thống không can thiệp)

- Người chơi nhìn thấy thông tin của nhau trong phòng.

- Chủ phòng được phép bắt đầu trận đấu. hệ thống đếm ngược từ 10 - 0 để người chơi chuẩn bị

- Người chơi có thể rời khỏi phòng. Nếu chủ phòng rời khỏi phòng. Người vào kế tiếp sẽ là chủ phòng. Nếu người cuối cùng rời phòng thì phòng bị xóa

- Không thể tham gia vào phòng nếu phòng đó đang chơi hoặc đã chơi xong.

- Mỗi người chơi điều khiển một con rắn.

- Cho phép lựa chọn bản đồ khi tạo phòng

- Rắn tăng trưởng kích thước khi ăn mồi

- Đầu rắn chạm vào thân của rắn khác (hoặc thân của nó) sẽ chết

- Người chơi điều khiển rắn tồn tại đến cuối cùng sẽ thắng.

**+ Server:**

- Chạy chương trình server

- Chọn địa chỉ url hợp lệ

- Nhấn vào nút khởi động server để server xử lý, nếu đúng thì chấp nhận và server ở trạng thái running chờ đợi hành động từ client

- Nhận yêu cầu từ client nếu thỏa mãn và hợp lệ thì phê duyệt

**+ Client:**

- Chạy chương trình phía client khi server đã ở trạng thái running thì kết nối đã hợp lệ và đúng

- Ở phía client người chơi bắt đầu thực hiện các thao tác khác để bắt đầu chơi, các hoạt động chơi được phía server xử lý

**+ Interface:**

- Lớp trung gian chứa các giao diện và chương trình quản lý trung gian giao tiếp, tương tác giữa client và server

## Thiết kế chương trình

- Client:

+ Lớp JFrameClient giao diện phía client

+ Lớp JPanelGame giao diện Jpanel phần các con rắn hoạt động và chơi

- Interface: giao diện trung gian giao tiếp giữa server và client

+ Lớp BanDo quản lý các danh sách các bản đồ được đạo ra từ server

+ Lớp Diem xác định vị trí trên giao diện JPanelGame vị trí cụ thể (tọa độ) khi di chuyển của rắn

+ Lớp GiaoDienQLDSphongchoi gọi các hàm xử lý ở lớp QLDSPhongChoi ở Server khi chơi và quá trình người chơi tạo phòng, tham gia vào phòng đến khi kết thúc trò chơi

+ Lớp PhongChoi quản lý quá trình người chơi bắt đầu vào, và đang ở trong phòng chơi.

+ Lớp Ran quản lý con rắn, thể hiện độ dài, vị trí, trạng thái, tên rắn , id rắn.

- Server:

+ Sở hữu list các bản đồ để người dùng gọi đến :

+ Lớp BanDo01

+ Lớp BanDO02

+ Lớp BanDo03

+ Lớp BanDo04

+ Lớp BanDoCoDien

+ Lớp JFrameServer giao diện server

+ Lớp QLDSPhongChoi chứa các hàm xử lý khi chơi và quá trình người chơi tạo phòng, tham gia vào phòng đến khi kết thúc trò chơi.

# TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

## Giao diện

- Khởi động server:

Graphical user interface

Description automatically generated

1. Giao diện server

- Khởi động server thành công, server ở trạng thái running chờ kết nối

Graphical user interface, text, application

Description automatically generated

1. Khởi động server thành công

- Khởi động client

A picture containing graphical user interface

Description automatically generated

1. Giao diện client

- Nhập tên cho người chơi đầu tiên là “chu phong day nha” rồi tạo phòng, giao diện xuất hiện một mã code để người khác dùng tham gia.

- Khi đã tạo phòng thành công thì con rắn màu “ xanh dương “ đại diện cho chủ phòng xuất hiện

Graphical user interface

Description automatically generated with low confidence

1. Giao diện client khi chủ phòng tạo phòng thành công

- Khi một người chơi khác nhập mã code “1959” do chủ phòng thông báo, sau đó bấm tìm phòng thì vào phòng chơi. Con rắn màu vàng đại diện cho người chơi tham gia vào xuất hiện.

A picture containing graphical user interface

Description automatically generated

1. Giao diện client khi người chơi tham gia vào phòng của chủ phòng thành công

- Bắt đầu chơi thì “ mồi màu trắng” xuất hiện

A picture containing graphical user interface

Description automatically generated

1. Tạo phòng thành công những người chơi bắt đầu chơi

- Giao diện server khi người chơi đang chơi trong phòng

Graphical user interface, text, application

Description automatically generated

1. Giao diện server khi người chơi đang chơi

- Giao diện các bản đồ người dùng có thể tự chọn:

+ Giao diện bản đồ cổ điển:



1. Giao diện bản đồ cổ điển

+ Giao diện bản đồ BanDo01:

Shape, rectangle

Description automatically generated

1. Giao diện bản đồ BanDo01

+ Giao diện bản đồ BanDo02:

A picture containing chart

Description automatically generated

1. Giao diện bản đồ BanDo02

+ Giao diện bản đồ BanDo03:

A picture containing logo

Description automatically generated

1. Giao diện bản đồ BanDo03

+ Giao diện bản đồ BanDo04:

A picture containing shape

Description automatically generated

1. Giao diện bản đồ BanDo04

+ Giao diện client khi 4 người cùng tham gia vào phòng

Graphical user interface, application

Description automatically generated

1. Giao diện client của người chơi tkhi 4 người cùng tham gia vào một phòng

+ Khi phòng đã đủ 4 người và có thêm người muốn vào phòng:

Graphical user interface, application

Description automatically generated

1. Cảnh báo phòng đã đủ 4 người chơi

+ Màn hình của 4 người chơi trong phòng:

Graphical user interface, application

Description automatically generated

1. Màn hình 4 người chơi cùng lúc trong phòng

+ Khi người chơi có tên “abcde” tham gia vào phòng nhập sai mã code hiển thị thông báo kiểm tra lại mã code

Graphical user interface, application

Description automatically generated

1. Khi người chơi tham gia vào phòng và nhập sai mã code

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

### Nhận xét

* + Ưu điểm:
    - Trong thời gian học tập, tìm hiểu về lý thuyết, công nghệ và áp dụng công nghệ nhóm chúng em đã đạt được những kết quả sau đây:

-Về mặt lý thuyết cơ bản đã sử dụng được kỹ thuật RMI ( Remod method invocation) trong đề tài

- Về mặt thực tiễn ứng dụng, đã có thể triển khai game cho nhiều người chơi cùng một lúc

- Giao diện thân thiên, dễ nhìn, dễ sử dụng

* + Nhược điểm:
    - Trong thời gian ngắn và lần đầu tiếp xúc với công nghệ mới và đề tài còn mới mẻ với chúng em. Nên chúng em vẫn còn nhiều hạn chế và thiếu sót như sau:

- Giao diện còn đơn giản, chưa đẹp

- Chỉ sử dụng được kỹ thuật RMI ở mức cơ bản. Mặc dù sử dụng kỹ thuật RMI là kỹ thuật cho phép nhiều người chơi ở các máy khác nhau và chạy trên môi trường mạng. Tuy nhiên, trong đề tài này chỉ triển khai được trên một máy

- Chỉ mới xử lý được game cho tối đa 4 người chơi cùng một lúc, chưa thể xử lý cho nhiều người chơi hơn

## Hướng phát triển

* + Một số hướng nghiên cứu và phát triển đề tài như sau:
    - Xây dựng giao diện đẹp hơn, thân thiện hơn và dễ sử dụng hơn
    - Phát triển game có thể chơi nhiều nhiều người nhất có thể và ở các máy khác nhau
    - Đưa chương trình chạy trên web và app để ứng dụng rộng rãi hơn
    - Cải tiến kỹ thuật code tối ưu hơn, sạch hơn, và dễ đọc hơn và tái sử dụng tốt hơn

1. LẬP TRÌNH MẠNG

Đề tài: Xây dựng chương trình client tương tác với Web API.

# GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

## NỘI DUNG ĐỀ TÀI

Xây dựng chương trình quản lí học sinh, bài học của trung tâm dạy lập trình cho trẻ em.

## MỤC TIÊU

Tìm hiểu cơ chế hoạt động của Winform, sử dụng thư viện để gọi Rest API, tìm hiểu giao thức HTTP, REST API.

## MÔI TRƯỜNG SỬ DỤNG

Chương trình client: chương trình Winform được viết bằng ngôn ngữ C#.

Chương trình server: WebAPI dùng framework của C# (ABP framework).

Trong đồ án môn học này, em sử dụng phần mềm Visual Studio, Visual Studio Code để phát triển ứng dụng.

# CƠ SỞ LÝ THUYẾT

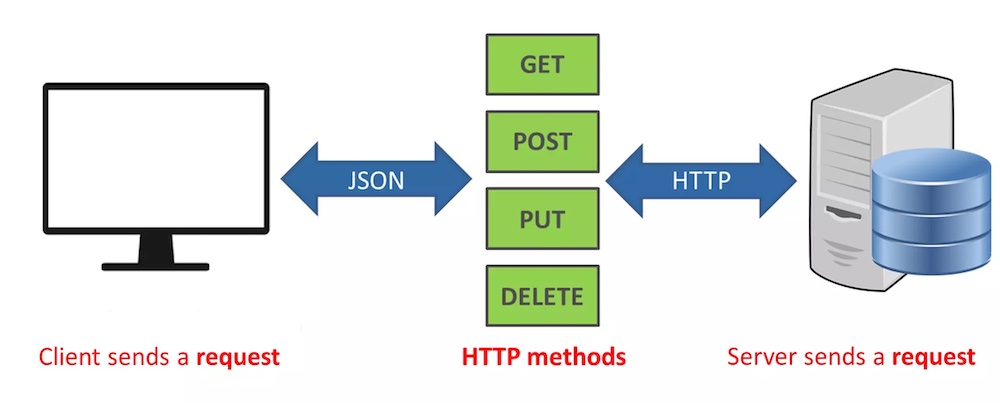
Nhờ sự phát triển của công nghệ thông tin nói chúng và Internet nói riêng, nhiều công việc, tác vụ của con người đã có thể thực hiện một cách dễ dàng hơn. Trong đó, vấn đề quản lí dữ liệu dạy học của các trung tâm dạy học bằng các công cụ truyền thống như excel, word,.. đang được chuyển dần thành các ứng dụng cụ thể cho từng trung tâm.

Trong bài đồ án này sẽ tập trung tìm hiểu về một ứng dụng desktop trong quản lí thông tin của một trung tâm dạy học: học sinh, các lớp học, bài học,… bằng Winform.

## REST API:

### Khái quát:

REST API (còn được biết với tên gọi RESTful API) là một giao diện lập trình ứng dụng (API) tuân thủ các ràng buộc và quy ước kiến trúc REST được sử dụng trong việc giao tiếp giữa client và server. REST là viết tắt của REpresentational State Transfer, nó được tạo ra bởi nhà khoa học máy tính Roy Fielding.



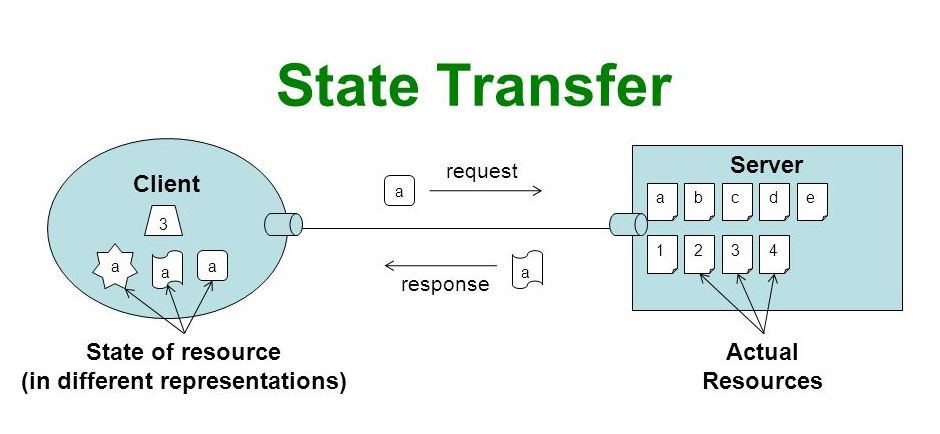
1. Rest API – Giao tiếp client và server

REST API thường vẫn sử dụng giao thức HTTP/1 kèm theo các định nghĩa trước đó mà cả client và server cần tuân thủ.

Hiện tại, các REST API dùng JSON rất phổ biến. Một ít có thể vẫn còn dùng XML.

### Hai thành phần trong REST API

**REST** (REpresentational State Transfer) là một đại diện cho sự chuyển đổi dữ liệu. Trong kiến trúc này client và server hoàn toàn độc lập, chúng không biết gì về nhau. Mỗi một request REST API đều không mang theo trạng thái trước đó (stateless). Như vậy để đôi bên trao đổi state, chúng sẽ phải thông qua các resources. Các resource này chính là phần đại diện cho sự thay đổi dữ liệu.

**API** (Application Programming Interface) là giao diện lập trình ứng dụng. Giao diện này không dành cho người dùng cuối mà dành cho các nhà phát triển (developer). Nó là cái “bề mặt”, chỉ thấy được phần khai báo (tên, tham số, kiểu trả về. 

1. Quá trình chuyển email

### Request và Response trong REST API

#### Phương thức

Như đã đề cập ở trên, để trao đổi state chúng sẽ cần giao tiếp resource thông qua việc gởi các request response thông qua HTTP/1. Cụ thể việc giao tiếp này là thế nào thì chúng cần chỉ định các method tương ứng bao gồm:

**GET**: Trả về một Resource hoặc một danh sách Resource.

**POST**: Tạo mới một Resource.

**PUT**: Cập nhật thông tin cho Resource (toàn bộ resource).

**PATCH**: Cật nhật thông tin cho resourse (một phần resource).

**DELETE**: Xoá một Resource.

#### Header: Authentication và quy định kiểu dữ liệu trả về

Mỗi một request không hề biết bất kỳ thông tin gì trước đó. Khác với khi chúng ta truy cập web, trình duyệt sẽ có session và cookie để hỗ trợ phân biệt request đấy là của ai, thông tin trước đó là gì.

Trong REST, nếu một request cần xác thực quyền truy cập, chúng sẽ phải dùng thêm thông tin trong header. Ví dụ như thông tin Authorization sẽ mang theo một user token. Hiện có 3 cơ chế Authentication chính:

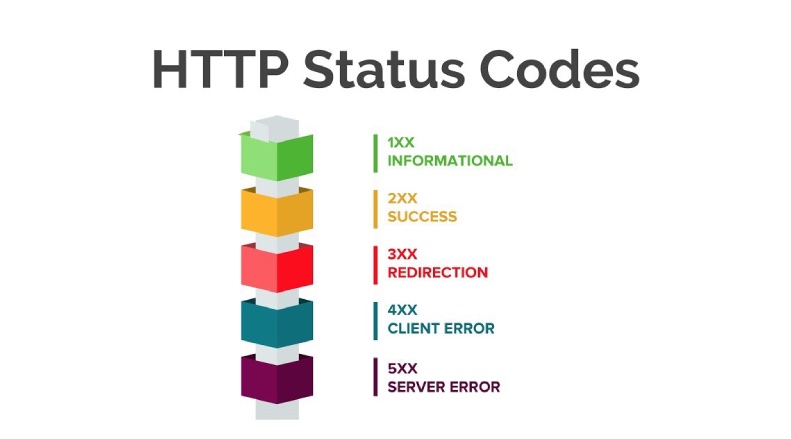
* HTTP Basic
  + - * JSON Web Token (JWT)
      * OAuth2

Ngoài ra Header còn giúp client chỉ định được loại content cần trả về từ server – content type. Việc này được thực hiện thông qua phần Accept trong header. Giá trị của nó thường là MIME type:

* image — image/png, image/jpeg, image/gif
* audio — audio/wav, audio/mpeg
* video — video/mp4, video/ogg
* application — application/json, application/pdf, application/xml, application/octet-stream

### Status Code

Response trong REST API sẽ bao gồm một status code quy định cụ thể từng trường hơp.



1. Ý nghĩa status code

Một số status phổ biến:

* 200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.
* 201 Created – Trả về khi một Resouce vừa được tạo thành công.
* 204 No Content – Trả về khi Resource xoá thành công.
* 304 Not Modified – Client có thể sử dụng dữ liệu cache, resource server không đổi gì.
* 400 Bad Request – Request không hợp lệ
* 401 Unauthorized – Request cần có xác thực.
* 403 Forbidden – bị từ chối không cho phép.
* 404 Not Found – Không tìm thấy resource từ URI
* 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
* 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
* 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
* 422 Unprocessable Entity – Dữ liệu không được xác thực
* 429 Too Many Requests – Request bị từ chối do bị giới hạn

## HTTP/HTTPs

### Khái quát

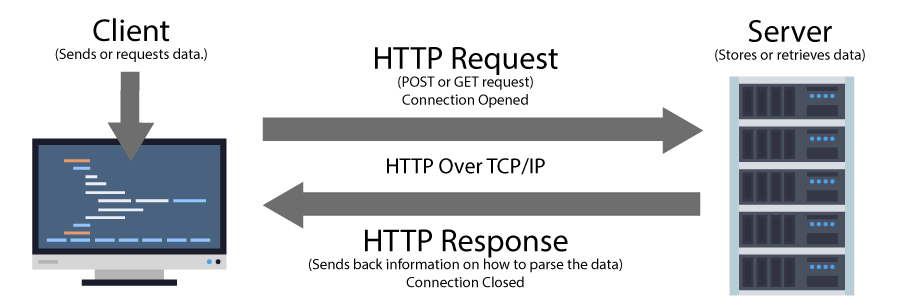
**HTTP** (HyperText Transfer Protocol) là giao thức truyền tải siêu văn bản được sử dụng trong www dùng để truyền tải dữ liệu giữa Web server đến các trình duyệt Web và ngược lại. Giao thức này sử dụng cổng 80 (port 80) là chủ yếu.

**HTTPS** (HyperText Transfer Protocol Secure) là giao thức Http có sử dụng thêm SSL (Secure Sockets Layer) để mã hóa dữ liệu trong lúc truyền tải dữ liệu nhầm gia tăng thêm tính an toàn cho việc truyền dữ liệu giữa Web server và trình duyệt Web. Giao thức Https thì sử dụng cổng 433 để truyền dữ liệu

### HTTP request và response

Sử dụng HTTP hoặc HTTPS để giao tiếp đọc thông tin từ máy chủ Web hoặc gửi thông tin đến máy chủ Web bằng trình duyệt Web.

* Giao tiếp HTTP được thực thi khi máy khách gửi một request đến máy chủ và máy chủ xử lý request rồi trả về một response cho nó.
* Các request và response HTTP về cơ bản được trao đổi dưới dạng văn bản. Ngoài thông tin dạng văn bản, thông tin nhận được từ máy chủ cũng có thể lưu trữ dữ liệu nhị phân như là hình ảnh...



1. HTTP request và response

## Abp framework

### Khái quát

ABP Framework cung cấp một kiến ​​trúc được đánh giá cao để xây dựng các giải pháp phần mềm doanh nghiệp với các phương pháp hay nhất trên nền tảng .NET và ASP.NET Core. Nó cung cấp cơ sở hạ tầng cơ bản, mẫu khởi động sẵn sàng sản xuất, mô-đun, chủ đề, công cụ, hướng dẫn và tài liệu để triển khai kiến ​​trúc đó đúng cách và tự động hóa các chi tiết và công việc lặp đi lặp lại nhiều nhất có thể.

### Mô hình

ABP cung cấp một kiến ​​trúc phần mềm hoàn chỉnh, theo mô-đun và nhiều lớp dựa trên các nguyên tắc và mẫu Domain Driven Design. Nó cũng cung cấp cơ sở hạ tầng cần thiết để thực hiện kiến ​​trúc này.

### Application Modules

Mô-đun ứng dụng cung cấp các chức năng ứng dụng dựng sẵn:

**Tài khoản(**[**Account**](https://docs.abp.io/en/abp/latest/Modules/Account)**):** Cung cấp giao diện người dùng để quản lý tài khoản và cho phép người dùng đăng nhập/đăng ký vào ứng dụng.

**Danh tính(**[**Identity**](https://docs.abp.io/en/abp/latest/Modules/Identity)**)**: Quản lý các đơn vị tổ chức, vai trò, người dùng và quyền của họ, dựa trên thư viện Microsoft Identity.

[**OpenIddict**](https://docs.abp.io/en/abp/latest/Modules/OpenIddict): Tích hợp với OpenIddict.

**Quản lý đối tượng thuê(**[**Tenant Management**](https://docs.abp.io/en/abp/latest/Modules/Tenant-Management)): Quản lý đối tượng thuê cho ứng dụng nhiều đối tượng thuê (SaaS).

## Refit

### Khái quát

RESTful API là một tiêu chuẩn được sử dụng trong việc thiết kế API cho các phần mềm, ứng dụng và dịch vụ web để tạo sự thuận tiện cho việc quản lý các resource. Các tài nguyên hệ thống như tệp văn bản, ảnh, video, âm thanh hay dữ liệu di động là mục tiêu mà nó hướng tới, bao gồm các trạng thái tài nguyên được định dạng và truyền tải qua HTTP.

Hay có thể nói theo cách khác, nó sẽ là 1 cây cầy để bắc qua nếu các trang web thống nhất với nhau về việc trả lại dữ liệu tương ứng. Nhưng mà trăm nghe không bằng mắt thấy nên ở bài viết này mình sẽ không đi sâu về mặt lý thuyết, chỉ nêu tổng quát cách hoạt động của RESTful => từ đó đưa ra demo để có thể quan sát tốt hơn.

### Các dạng Request của Restful API

Http Method gồm có 9 loại nhưng RESTful chỉ sử dụng 4 loại phổ biến

GET (SELECT): Trả về một Resource hoặc một danh sách Resource.

POST (CREATE): Tạo mới một Resource.

PUT (UPDATE): Cập nhật thông tin cho Resource.

DELETE (DELETE): Xoá một Resource.

=> Tương ứng với cái tên thường gọi là CRUD (Create, Read, Update, Delete)

### Ưu điểm

Giúp cho ứng dụng trở nên rõ ràng hơn.

REST URL đại diện cho resource chứ không phải là hành động.

Dữ liệu được trả về với nhiều định dạng khác nhau như: xml, html, rss, json …

Code đơn giản và ngắn gọn.

REST chú trọng vào tài nguyên hệ thống.

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## BÀI TOÁN VÀ YÊU CẦU

### Bài toán:

Xây dựng ứng dụng desktop trong quản lí thông tin của một trung tâm dạy học: học sinh, các lớp học, bài học,… bằng Winform.

* Yêu cầu:
  + Quản lí user:
    - Admin
    - Giáo viên
  + Quản lí học sinh:
    - Thông tin học sinh, phụ huynh: Thông tin học sinh, phụ huynh.
    - Lớp học đang tham gia: Bài học đã học, thông tin đánh giá.
  + Quản lí Khoá học:
    - Thông tin Khoá học: Tên, mô tả, số buổi học, danh sách bài học.
    - Các chức năng CRUD cơ bản.
    - Một khoá học có thể dung cho nhiều lớp học
  + Quản lí lớp học:
    - Thông tin lớp học: tên, thời gian, khoá học, học phí…
    - Các chức năng CRUD cơ bản.
    - Mỗi lớp học thuộc 1 kì học.
    - Mỗi lớp học sử dụng 1 khoá học
  + Quản lí kì học:
    - Một kì học sẽ có nhiều lớp học
    - Thông tin lớp học: thời gian, tên kì học
  + Quản lí buổi học:
    - Một lớp học có nhiều học sinh của nhiều lớp học khác nhau.
    - Tạo 1 buổi học: Chọn thời gian => chọn học sinh => chọn bài học cho học sinh.

## THIẾT KẾ CHƯƠNG TRÌNH

### Server API

#### Sơ đồ trình tự tổng quát

Diagram

Description automatically generated

1. Sơ đồ trình tự tổng quát

1. Người dung truy cập vào ứng dụng winform, thực hiện đăng nhập.

2. Client-winform sẽ thực hiện gọi 1 yêu cầu đăng nhập đến server.

3. Nếu đăng nhập thành công server sẽ trả về 1 token. Dựa vào token này để định danh và phân quyền người dùng.

4. Người dùng thực hiện 1 hành động (thêm, sửa, xoá, lấy dữ liệu).

5. Client-winform gọi 1 API tương ứng với hành động của người dùng đến server.

6. Server thực hiện lấy dữ liệu hoặc cập nhật dữ liệu trên database sau đó trả về cho client kết quả(thường là 1 json, HTTP code)

7. Client nhận phản hồi từ server và hiển thị ra màn hình cho người dùng.

#### Cấu trúc solution

Diagram

Description automatically generated

1. Cấu trúc solution

Server sử dụng ABP framework. Cầu trúc solution gồm:

* + **.Domain.Shared**: Chứa các hằng số, enum,.. dùng chung cho cả solution, tất cả các phần đều có thể truy cập được đến nó.
  + **.Domain**: Chứa các entity của dự án, các interface của repository. Chỉ .Application có thể gọi đến domain.

Text

Description automatically generated

1. Class entity
   * **.EntityFrameworkCore**: triển khai các interface được khai báo ở Domain. Các thao tác truy cập đến database được sử dụng ở đây thông qua dbcontext
   * Text

     Description automatically generated
2. Class repository
   * **.Application.Contract**: chứa các DTO(Data Transfer Object), các interface service.

Text

Description automatically generated

1. Class DTO

**.Application**: triển khai các interface được khai báo ở .Application.Contract. Chứa các service tương ứng với các API cần thiết. Gọi đến các repository được triển khai ở EntityFrameworkCore thông qua interface ở .Domain (sử dụng dependence injection pattern) để thực hiện lấy dữ liệu hoặc cập nhật dữ liệu.

Text

Description automatically generated

1. ClassAppService
   * .**Web:** Chứa các User Interface của web, appsettings(chứa chuỗi kết nối và các cấu hình cần thiết).

#### File appsettings.json

Text

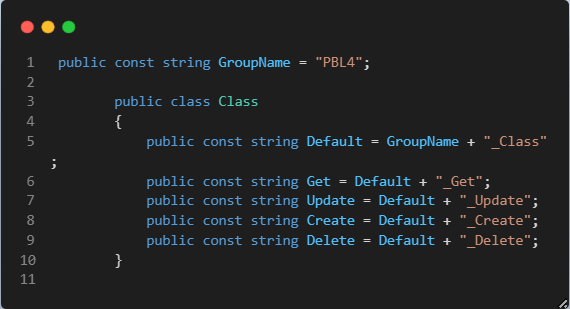
Description automatically generated

1. File appsettings.json
   * Địa chỉ của Server được chạy ở local: <https://localhost:44313>
   * Chuỗi kết nối CSDL: "Server=localhost;Database=PBL4;Trusted\_Connection=True"
   * Chuỗi xác thực người dung: "https://localhost:44313"

#### Xử lí đăng nhập và phân quyền

Sử dụng user đăng nhập mặc định của ABP Frame work và phân quyền có sẵn.

Ví dụ quyền truy cập lớp học:



Text

Description automatically generated

1. Xác định quyền truy cập

Text

Description automatically generated

1. Sử dụng các quyền truy cập ở appservice

#### Thiết kế cơ sở dử liệu

Graphical user interface, application, Word

Description automatically generated

#### Deployment

Sử dụng Docker để deploy server trên Digitalocean.

Docker file:

Text

Description automatically generated

Server sử dụng DigitalOcean: Server được deploy ở địa chỉ <http://143.198.84.239/>

### Client- winform

#### Các thư mục cần thiết

* + **BaseFunction**: chứa template class để gọi API
    - Text

      Description automatically generated

1. Tempalte class GetAPIByService để gọi API

**Token** được lấy lúc đăng nhập và lưu ở biến ConfigManger.Token:

Text

Description automatically generated

1. Get Token from server
   * **Dto**: chứa các entity.
   * **SdkCommon:** chứacác interface tương ứng với các function để gọi API

**Text

Description automatically generated**

1. Interface trong SDK common
   * Ví dụ gọi 1 API:

Text

Description automatically generated

1. Gọi API

classApi.GetAsync(id) tương ứng với API có url: “BaseUrl/api/app/class/{id}”

classDto sẽ chứa các thông tin của class có id là id. Sau đó hiển thị các thông tin lên view.

* + Xử lí ngoại lệ
  + Text

    Description automatically generated

1. Xử lí ngoại lệ
   * + Khi khởi tạo chương trình gọi

**Application.SetUnhandledExceptionMode(UnhandledExceptionMode.CatchException);**

để hiện thông báo khi có lỗi xảy ra

# TRIỂN KHAI VÀ ĐÁNH GIÁ KẾT QUẢ

## Giao diện

Khởi chạy chương trình Winform trên phần mềm Visual Studio.

Có thể sử dụng WebAPI là server đã được deploy hoặc chạy chương trình ở local.

Nếu sử dụng server đã deploy chỉnh sửa file appsetting ở winform như sau:

http://143.198.84.239

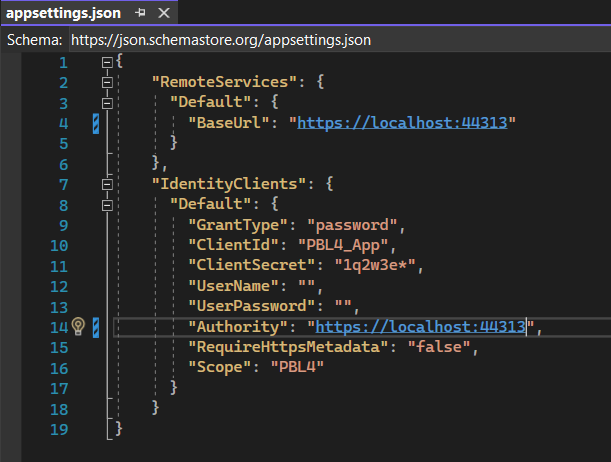
Text

Description automatically generated

1. Appsetting server

Nếu dùng server local:

https://localhost:44313



1. Appsetting local

Giao diện khi khởi động cho phép đăng nhập hoặc truy cập với quyền khách.

Tài khoản:

* + Admin:
    - username: admin
    - password: 1q2w3E\*
  + Giáo viên:
    - username: nham
    - password: 1q2w3E\*
  + Phụ huynh
    - Username: vy
    - Password: 1q2w3E\*

Graphical user interface

Description automatically generated

1. Giao diện login

Giao diện màn hình chính khi truy cập:

Các màn hình có các chức năng cơ bản: Tìm kiếm, thêm, sửa, xoá.

Table

Description automatically generated

1. Giao diện chính

Thông tin chi tiết của 1 học sinh: cho phép xem thông tin các lớp học mà học sinh tham gia

Graphical user interface, application

Description automatically generated

1. Chi tiết, chỉnh sửa thông tin học sinh

Chi tiết của 1 khoá học: cho phép thêm và xoá 1 bài học khỏi dành sách bài học của khoá.

Graphical user interface, application

Description automatically generated

1. Chi tiết khoá học

Chi tiết bài học: Bao gồm link tài liệu, thông tin hướng dẫn cho buổi học

Graphical user interface, application

Description automatically generated

1. Chi tiết bài học

Thông tin lớp học: cho phép chọn kì học cho lớp học, mỗi lớp học chỉ thuộc 1 kì học.

Graphical user interface

Description automatically generated

1. Chi tiết lớp học

Thông tin của 1 buổi học: Cho phép chọn học sinh tham gia buổi học và chọn bài học cho từng học sinh.

Graphical user interface, application

Description automatically generated

1. Chi tiết buổi học

Màn hình swagger hiển thị dành sách các API có thể sử dụng và có thể thử trực tiếp trên đó.

Graphical user interface, text, application, email

Description automatically generated

1. Danh sách các API

Ví dụ gọi API để tìm kiếm lớp học trên swagger.

Graphical user interface, application

Description automatically generated

1. Thử gọi API trên swagger

Kết quả trả về số lượng lớp học trả về và thông tin chi tiết của lớp học dưới dạng json:

Text

Description automatically generated

1. Kết quả

Cập nhật các quyền hạn cho các tài khoản hoặc nhóm tài khoản thông qua web

Có thể them và chỉnh sửa quền cho các vai trò.

Graphical user interface, text, application, email

Description automatically generated

1. Danh sách các vai trò

Chỉnh sửa quyền cho vai trò

Graphical user interface, application

Description automatically generated

1. Chỉnh sửa quyền cho vai trò

Danh sách các người dùng, có thể chỉnh sửa thông tin và vai trò hoặc quyền riêng biệt cho từng người dung.

Graphical user interface, application, Teams

Description automatically generated

1. Quản lí người dùng

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Ý nghĩa đề tài:

### Ưu điểm

Trong thời gian học tập, tìm hiểu chúng em đạt được kết quả như sau:

-Áp dụng các kiến thức tìm hiểu được về API để tạo nên một ứng dụng có thể sử dụng được.

-Tìm hiểu 1 công nghệ mới (ABP framework) và áp dụng thành công trong đồ án.

-Sử dụng công cụ quản lí mã nguồn(git) và thiết kế database(dbbdiagram.io).

-Đã deploy thành công server trên môi trường internet và có thể truy cập ở mọi nơi.

### Nhược điểm

Tuy nhiên vẫn còn một số nhược điểm:

- Giao diện vẫn còn đơn giản.

- Ứng dụng chưa sử dụng trong thực tế do chưa thể deploy, chúng em đã thử nhưng đã thử nhưng đang gặp lỗi.

- Ứng dụng đang chỉ sử dụng trên máy tính. Trong thời gian tới chúng em sẽ tạo 1 phiên bản dành cho web để có thể sử dụng được trên mọi thiết bị.

## Hướng phát triển:

Tiếp tục phát triển ứng dụng, cập nhật lại giao diện. Trao đổi với 1 trung tâm cụ thể để sử dụng ứng dụng trong thực tể. Phát triển thêm web dựa trên nền Web API đã có, thêm các chức năng để đăng bài viết, đăng kí học thông qua web.

TÀI LIỆU THAM KHẢO

Hệ điều hành

[1] Sự khác biệt giữa RMI và Socket https://www.geeksforgeeks.org/difference-between-rmi-and-socket/

[2] RMI trong Java ứng dụng phân tán https://daynhauhoc.com/t/giao-thuc-rmi-trong-java-ung-dung-phan-tan/23963

[3] Mạng máy tính thầy Nguyễn Tấn Khôi

[4] Lập trình java thầy Mai Văn Hà

[5] Lập trình mạng thầy Mai Văn Hà

[6] Java RMI và ứng dụng phân tán đơn giản https://viblo.asia/p/java-rmi-va-ung-dung-phan-tan-don-gian-OEqGj50KM9bL

[7] Tìm hiểu về xử lý đa luồng trong java https://viblo.asia/p/tim-hieu-ve-xu-li-da-luong-trong-java-m68Z0xyQZkG

[8] Giới thiệu về java RMI https://viblo.asia/p/gioi-thieu-ve-java-rmiremote-method-invocation-XogBG2xrRxnL

[9] https://en.wikipedia.org/wiki/Java\_remote\_method\_invocation

[10] https://www.tutorialspoint.com/java\_rmi/java\_rmi\_application.htm

[11] https://www.javatpoint.com/RMI

Lập trình mạng

[1] Tìm hiểu API <https://aws.amazon.com/vi/what-is/api/>

[2] Tìm hiểu ABP: <https://docs.abp.io/en/abp/latest>

[3] Tìm hiểu Refit: https://code-maze.com/using-refit-to-consume-apis-in-csharp/

[4] Tìm hiểu Refit: https://viblo.asia/p/thiet-ke-restful-api-goi-api-bang-httpclient-trong-aspnet-gAm5yVxVKdb